KPCI-1812 数据采集卡
使用说明书

北京科瑞兴业科技有限公司

北京科瑞兴业科技有限公司　　地址：北京市海淀区知春里 28 号开源商务写字楼 212/213 室
邮政编码：100086　　电话：010-51650651　010-62527214　　传真：010-62657424
http://www.krxgk.com
Sales E-mail: sgq@krxgk.com　　Tech Support E-mail: lilanzhen007@126.com

目录

第一章 概述

1、 介绍
2、应用
3、性能和技术指标
4、软件支持

第二章 主要元件位置图、信号输出插座和开关跳线选择定义

1、主要元件布局图
2、短路套设置
3、信号输入输出插座定义
4、模拟信号输入连接方式及应注意的问题

第三章 函数模块调用说明

1．A/D 采集过程流程图
2．函数说明

第四章 KPCI-1812 A/D 卡的校准、保修和注意事项

1. 注意事项
2. 校准
3. 保修

第一章　概述

一、介绍

KPCI-1812 卡是一种基于 PCI 总线的数据采集卡，插在计算机内的任一 PCI 插槽中，可构成实验室、产品质量检测中心和大专院校等各种领域的数据采集、分析和数据处理系统。也可构成工业现场的过程监控系统。KPCI-1812 板上装有 12Bit 分辨率的 A/D 转换器。为您提供了差分 16 路/单端共地 32 路的模拟输入通道。A/D 输入信号范围：±5V、0~10V。

- **总线：**32 位 PCI 总线，支持 PCI2.2 协议，即插即用。
- **PCI 芯片：**FPGA 接口芯片设计。
- **转换速度和精度：**100KHz，12 位 A/D 转换器，通过率为 100KS/s。
- **AD 通道数：**32 通道单端，16 通道差分输入。
- **转换方式：**支持软件查询方式、中断方式，外触发方式。
- **缓存：4**K 深度的 FIFO 缓存。
- **软件支持：**提供动态链接库供编程使用，提供 VB,VC 编程示例。

二、性能及技术指标

2.1、PCI 局部总线性能

PCI 总线宽度 32 位，同步工作频率可达到 33MHz，最高传输速率为 132MB/S

能够完成自动配置，实现设备的即插即用

2.2、模拟信号输入部分

模拟通道数： 32 路单端输入或 16 路差分输入

硬件增益：1、2 倍或可调增益，可通过硬件跳线设定量程范围。

模拟输入电压范围：±5V、0～10V

模拟输入阻抗: 10MΩ

2.3、A/D 转换电路部分

A/D 分辨率：12Bit(4096)

非线性误差：±1LSB(最大)

转换时间：10us

系统测量精度：0.1%

2.4 A/D 采集方式及周期

三种采样触发方式：定时触发、软件触发和外触发，可由软件设置使用 FIFO 还是不使用 FIFO。

2.6、FIFO 存储器

深度： 4K Words

宽度： 12Bits

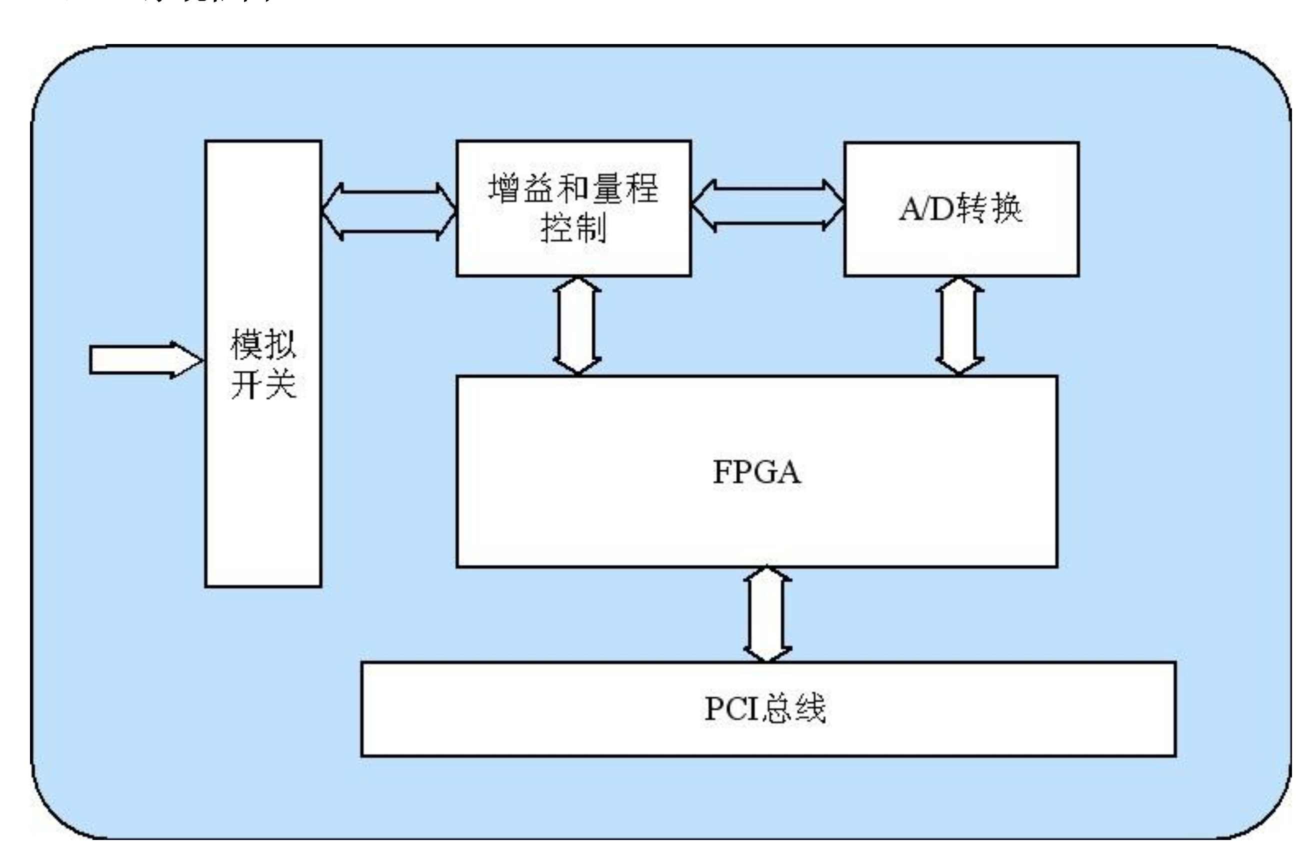
标志： 空、半满

三、软件支持

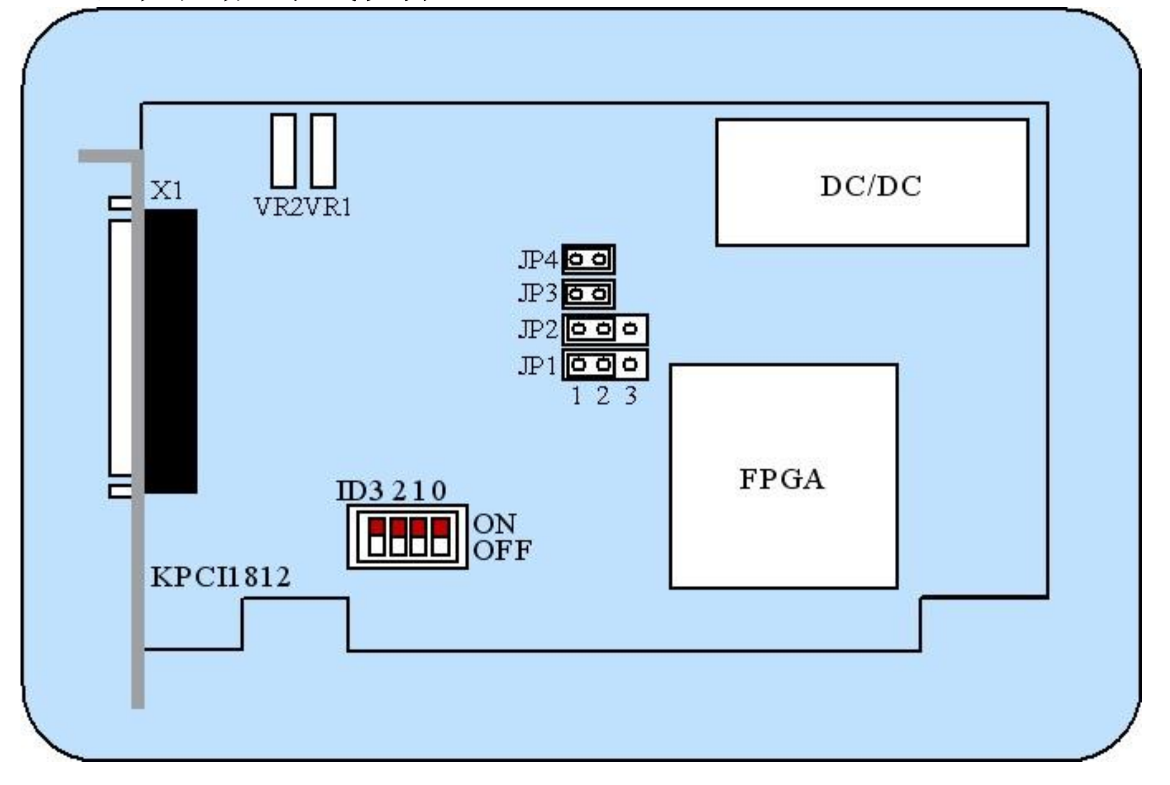
提供 Windows95/98/NT 下的多种语言的驱动，并提供在 VB 和 VC 环境下，开发的示例程序（详见软件说明部分）。

第二章 元件位置图、信号输出插座和开关跳线选择定义

一， 系统框图



二，量程和增益跳线说明



JP1 和 **JP2** 的 PIN2 和 PIN3 同时短接时设置量程为±5V, PIN2 和 PIN1 同时短接时设置

量程为 0~10V。

JP3，**JP4** 用于增益控制，JP3、JP4 都断开时为 1 倍增益，当短接 JP4 时为 2 倍增益控制，当短接 JP3 时为通过 R1 上焊接不同阻值的电阻实现任意增益控制。

三，A/D 转换调整用电位器说明

VR2：A/D 通道的满度调整电位器。
VR1：A/D 通道的调零电位器。

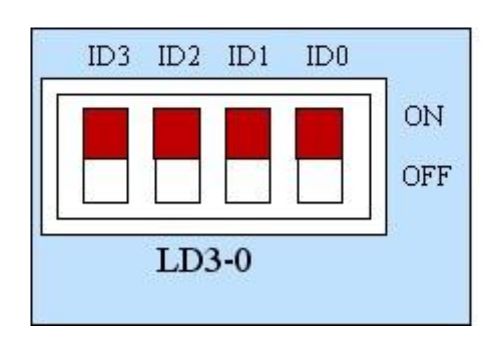
四、信号输入插座定义

4.1　模拟量输入接口信号定义（输入插座为DB37孔式弯针插座）

插座引脚号	引 脚 定 义	插座引脚号	引 脚 定 义
1	Ain1 (In1+)	20	Ain2 (In1-)
2	Ain 3 (In2+)	21	Ain4 (In2-)
3	Ain 5 (In3+)	22	Ain6 (In3-)
4	Ain 7 (In4+)	23	Ain8 (In4-)
5	Ain 9 (In5+)	24	Ain10 (In5-)
6	Ain11 (In6+)	25	Ain12 (In6-)
7	Ain13 (In7+)	26	Ain14 (In7-)
8	Ain15 (In8+)	27	Ain16 (In8-)
9	AGND	28	AGND
10	AGND	29	Ain17 (In9+)
11	Ain18 (In9-)	30	Ain19 (In10+)
12	Ain20 (In10-)	31	Ain21 (In11+)
13	Ain22 (In11-)	32	Ain23 (In12+)
14	Ain24 (In12-)	33	Ain25 (In13+)
15	Ain26 (In13-)	34	Ain27 (In14+)
16	Ain28 (In14-)	35	Ain29 (In15+)
17	Ain30 (In15-)	36	Ain31 (In16+)
18	Ain32 (In16-)	37	EXT_TRG
19	GND		

Ain 1～Ain 32：KPCI-1812 A/D卡单端模拟信号输入通道号
In1+ ～In16+ ：　差分模拟信号输入正端
In1- ～ In16- ：　差分模拟信号输入负端
AGND：模拟地
GND：数字地
EXT_TRG：外部触发信号（TTL电平），当EXT_TRG有一由低至高的变化(上升沿)时，KPCI-1812 A/D卡将按预先设定的采集通道总数进行采集，直至采集结束。程序举例见软件说明书相应部分。

用四位拨码开关 LD3-0 可用来设置板卡标识 ID，当用户在同一计算机中使用多块 KPCI-1812 构建自己的系统时，ID 设置功能极为有用。用户可方便的在硬件配置和软件编程过程中区分和访问每块采集卡。



ID3	ID2	ID1	ID0	Board ID
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

注意： ON 为 0 OFF 为 1

五、模拟输入信号的连接方式

5.1　单端输入方式：

KPCI-1812板均可按图5.1连接成模拟电压单端输入方式，32通道模拟输入信号连接到Ain 1～Ain 32端，其公共地连接到AGND端，注意：为了抑制噪声干扰，所有信号源接地端都应该一点接**AGND**。(见图5.1)

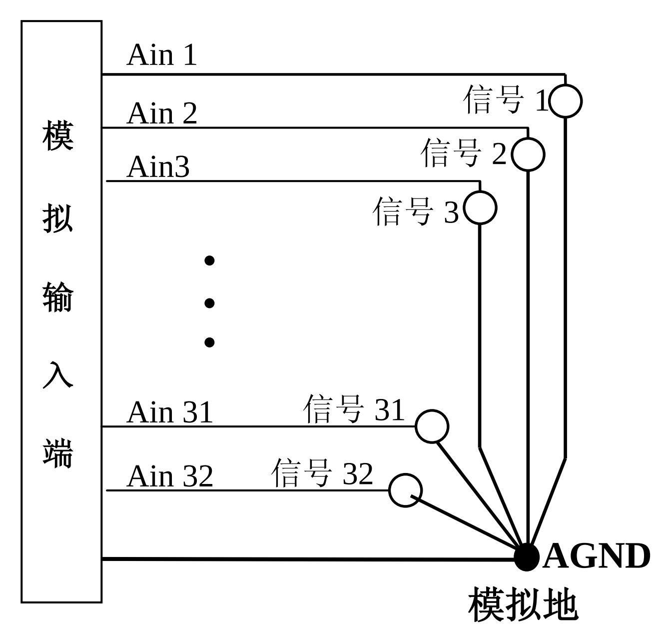


图 5.1 单端输入方式

5.2 差分输入方式：

KPCI-1812板可按图5.2连接成模拟电压差分输入方式，可以有效抑制共模干扰信号，提高采集精度。16通道模拟输入信号正端接到In1+～In16+端，其模拟输入信号负端接到In1-～In16-端，　(见图5.2)

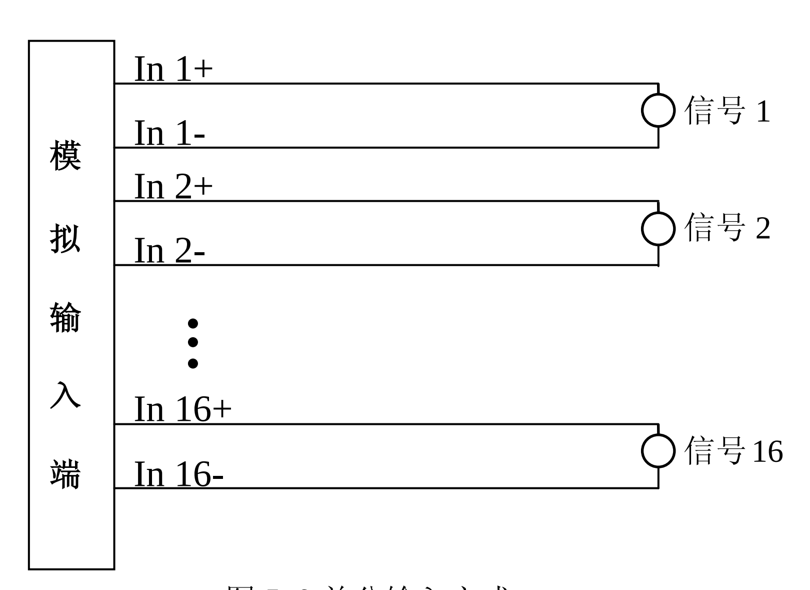


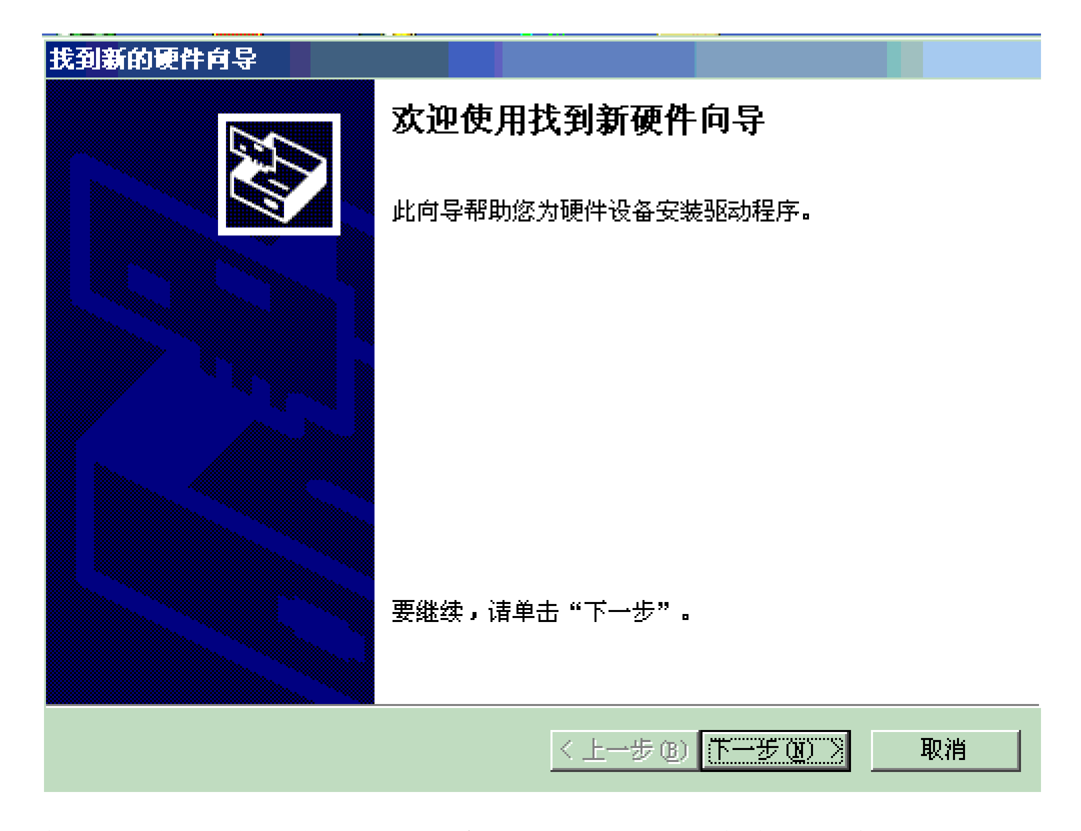
图 5.2 差分输入方式

第三章 KPCI-1812 设备驱动程序安装

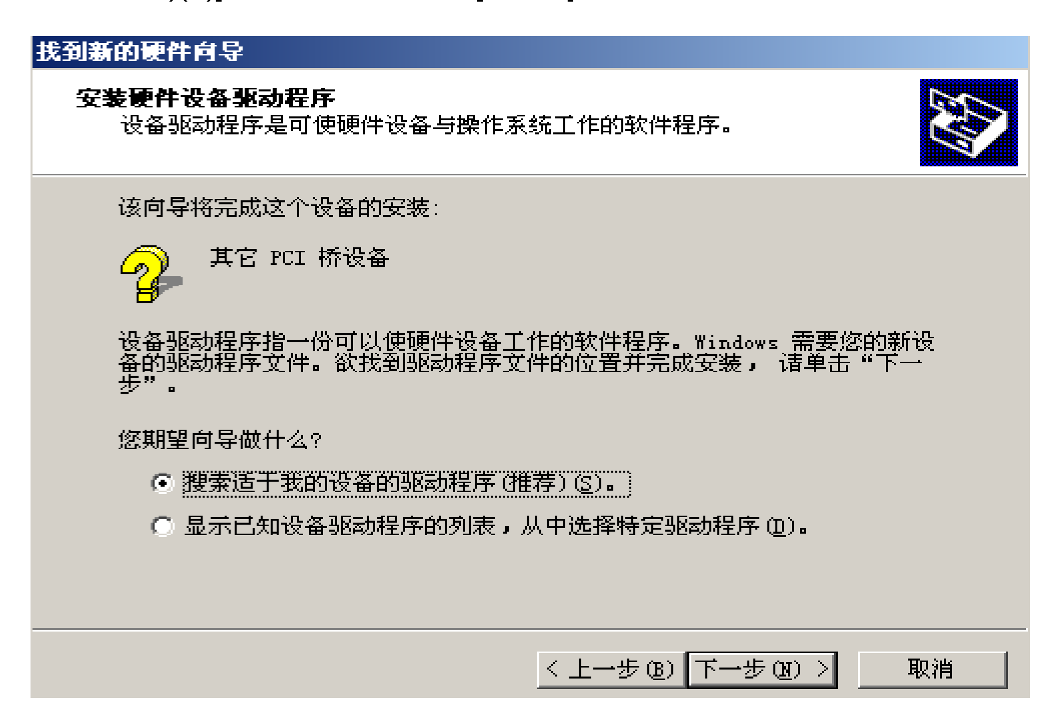
一、安装步骤

第一步 将 KPCI-1812 卡按硬件要求插入计算机主板上的任意一个 PCI 插槽中，并将其固定好，连接好其外接设备后，打开计算机电源，启动 Windows2000(XP)系统。

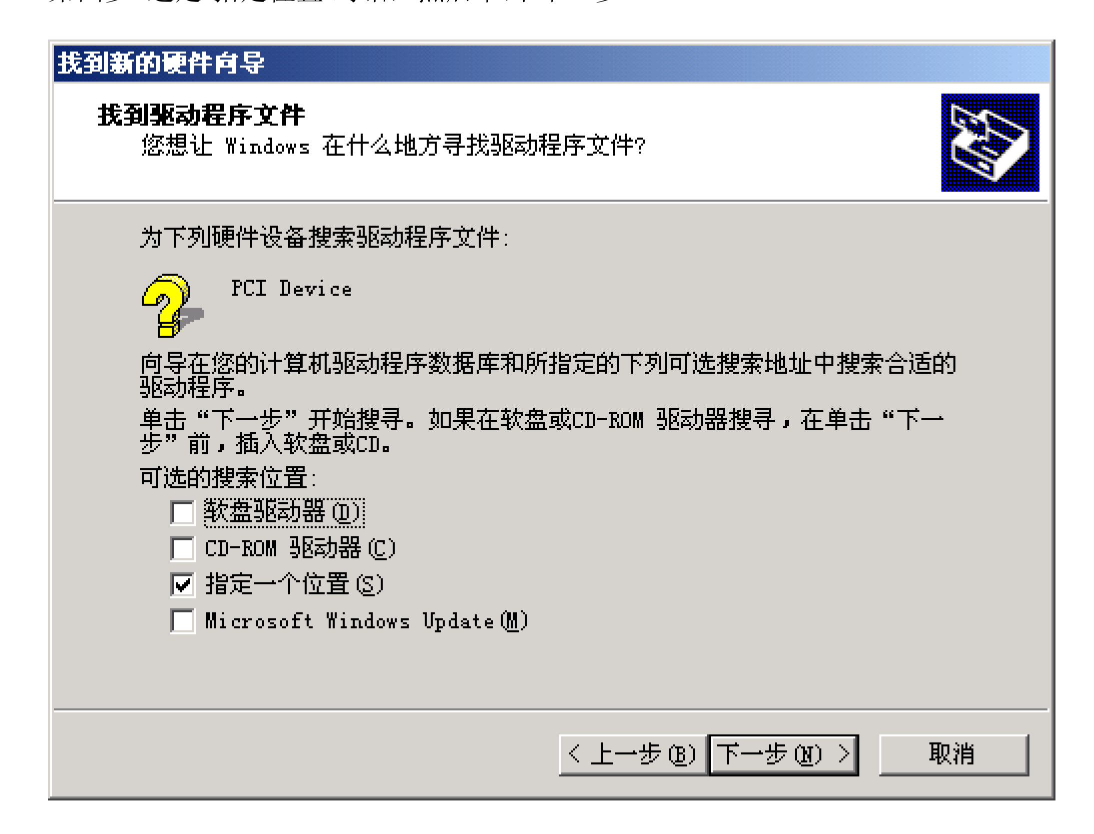
第二步 如果您正确地插好了 PCI 设备，Windows 系统在启动过程中便会发现这个新的 PCI 设备，并弹出 [欢迎使用找到新的硬件向导] 对话框，单击 [下一步] 按钮



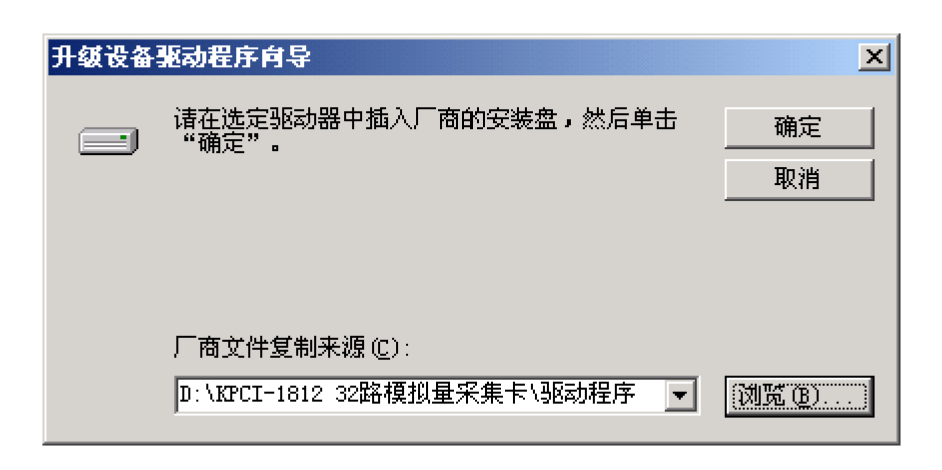
第三步 弹出[安装硬件设备驱动程序]对话框，在对话框中单击 [搜索适用于我的设备的驱动程序(推荐)(S)] 单选框，然后单击 [下一步] 按钮



第四步 选定“指定位置”对话，然后单击“下一步”。



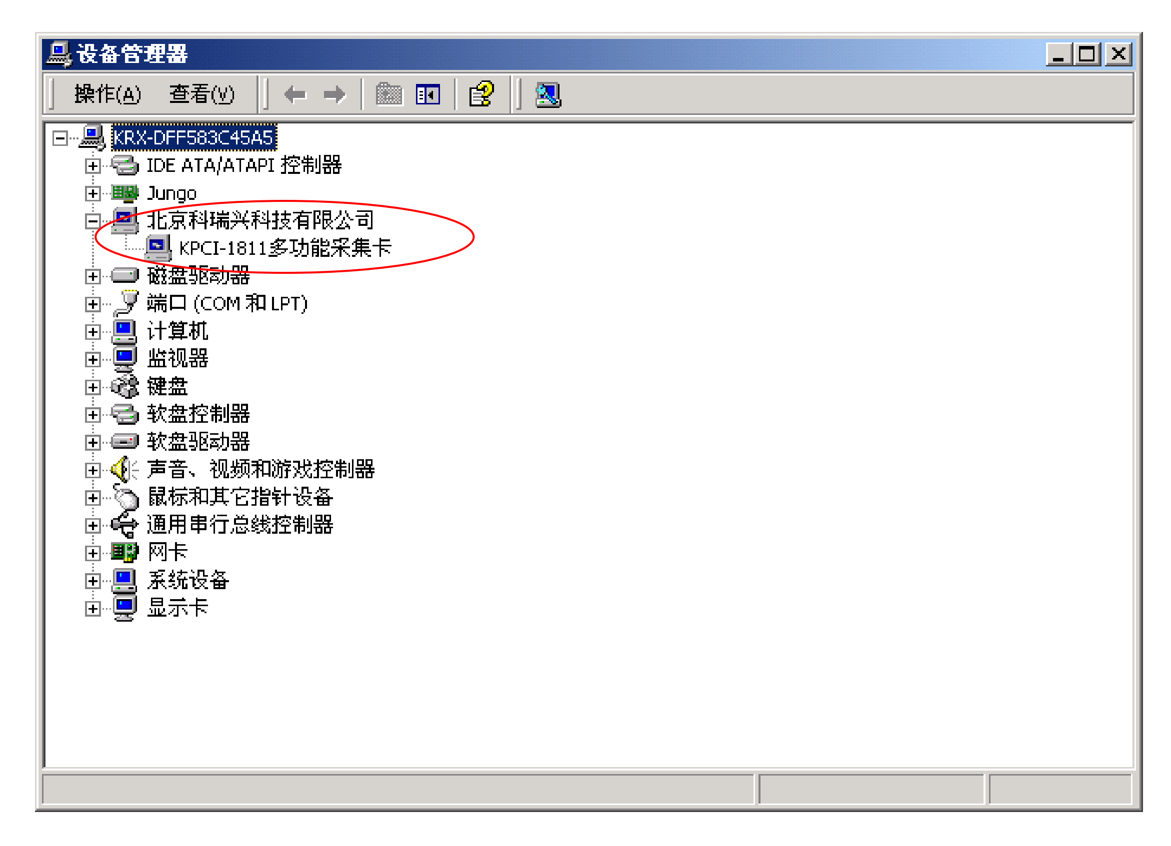
第五步 然后单击“浏览(R)”，将路径定位在光盘上的“PCI 总线测控板卡\KPCI-1812 32 通道模拟量输入卡\驱动程序\” 路径下，选择 KPCI812.inf 文件，然后单击“确定”按钮。



第六步 [找到新的硬件向导]，此步骤可能会出现 Windows 安装驱动程序的进度状态窗口，用户稍等片刻，然后出现[完成找到新硬件向导]对话框，单击 [完成]按钮。

二、安装结果验证

进入 Windows2000 [控制面板] 窗口，双击 [系统] 图标，弹出 [系统 特性] 对话框，在对话框中单击 [硬件] 标签页，然后单击 [设备管理器]按钮，进入 [设备管理器] 窗口，在 [本地计算机] 列表中检查是否有“北京科瑞兴业科技有限公司”显示，如下图所示。若有，表示 KPCI-1812 卡的驱动程序已成功安装，否则，说明您的安装过程出现了问题，请试着再安装，或向硬件供应商求助。



如果在 KPCI-1812 设备上出现黄色感叹号，可运行驱动程序文件夹中的 SETUP.bat，或把 KPCI1812S.SYS 文件拷贝到 windows\system32\drivers 目录下，重新扫描硬件。

第三节 PCI 设备软件测试系统的介绍

当您正确完成了第一或二节中的工作后，您便可以到光盘中“PCI 总线测控板卡\KPCI-1812 32 通道模拟量输入卡\编程” 文件夹，按照板卡的工作模式，运行程序，进行板卡测试。当用户自己编制应用程序时，可以参照光盘中“PCI 总线测控板卡\KPCI-1812 32 通道模拟量输入卡\编程” 文件夹中的编程示例程序。

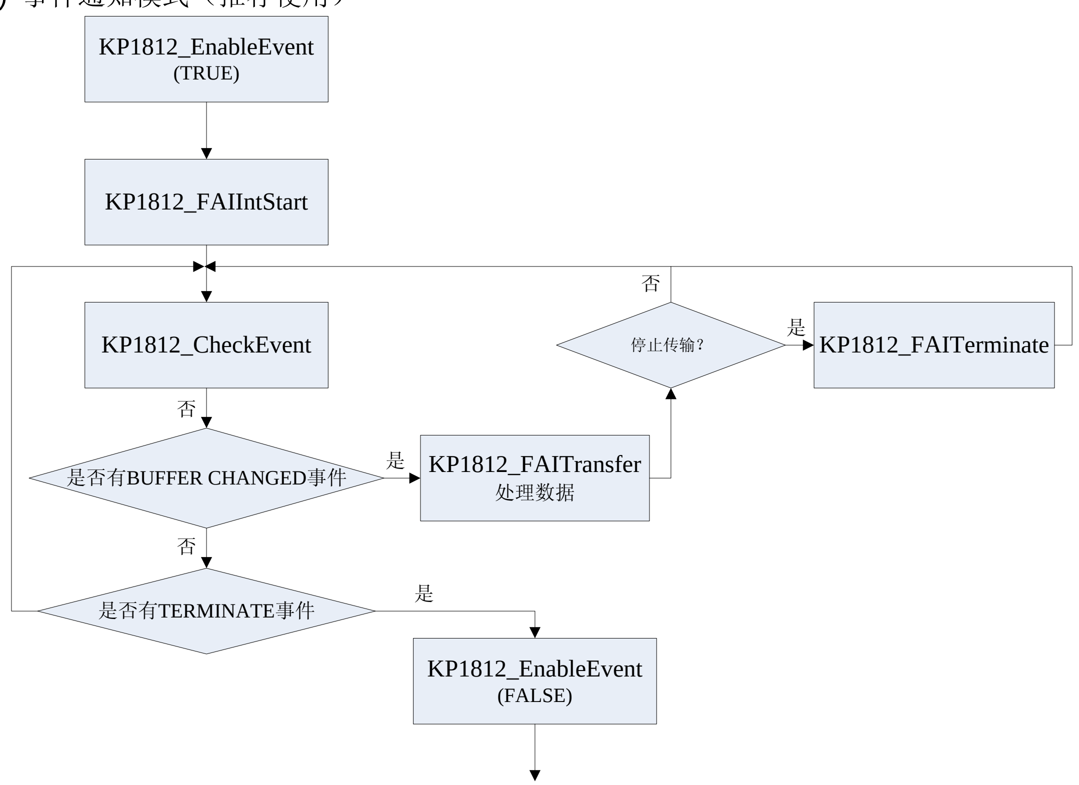
第四章 函数模块调用说明及编程实例

光盘中提供了两个分别在 VB 和 VC 下开发的示例程序，给出了 KPCI-1812 卡的各种触发方式、相关参数的设置过程和数据读取方法。对使用 VB 和 VC 的用户可以参照相应程序段根据实际需要利用函数库中提供的函数设计自己的软件，而对数据采集过程中不处理其他任务的用户，也可以直接利用示例程序完成数据。 初次使用动态链接库的用户还可以在程序中找到动态链接库的调用方法。为方便用户分析，示例程序以工程的形式提供了所有的资源和代码。用户在编译时需要注意重新指定动态连接库的路径，或把函数库拷到指定的位置。

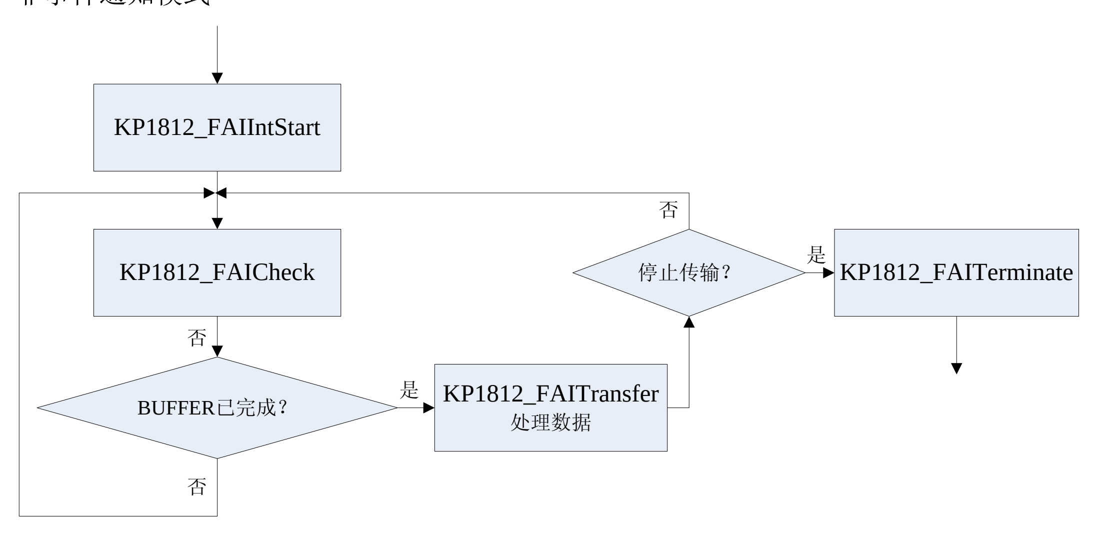
二、函数调用流程：

1、中断触发机制

a) 事件通知模式（推荐使用）



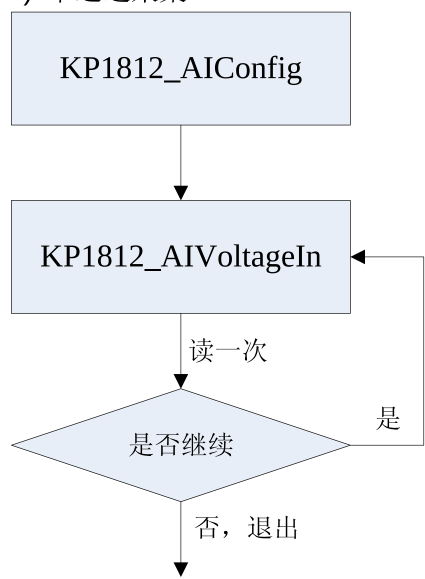
b) 非事件通知模式



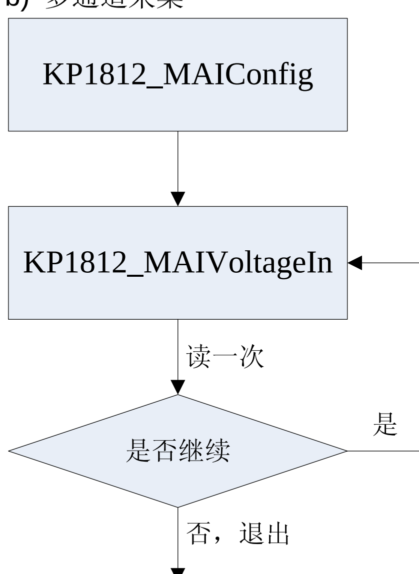
注意：以上流程图是中断触发的单通道的情况下数据采集的流程图，多通道则将KP1812_FAIIntStart改成KP1812_FAIIntScanStart

2、软件触发机制

a) 单通道采集



b) 多通道采集



二 函数说明

KPCI-1812 函数简表	
函数名	函数功能
KP1812_SelectDevice	选择要操作的设备
KP1812_SelectDeviceFromList	从列表中选择要操作的设备
KP1812_DeviceGetFeatures	获取设备特征
KP1812_DeviceGetProperty	获取设备指定的属性
KP1812_DeviceGetProperty	设置设备指定的属性
KP1812_AIConfig	配置指定的单个模拟量输入通道
KP1812_AIGetConfig	获取指定的单个模拟量输入通道配置
KP1812_AIBinaryIn	读取指定的单个通道模拟量输入的二进制值
KP1812_AIScale	转换二进制值为电压值
KP1812_AIVoltageIn	读取模拟量输入的电压值
KP1812_MAIConfig	配置指定的多个模拟量输入通道
KP1812_MAIBinaryIn	读取指定的多个通道模拟量输入的二进制值
KP1812_MAIVoltageIn	读取多个模拟量输入通道的电压值
KP1812_ReadPortByte	从指定I/O端口读一个字节的数据
KP1812_WritePortByte	写一个字节的数据的到指定的 I/O 端口
KP1812_ReadPortWord	从指定 I/O 端口读一个字的数据
KP1812_WritePortWord	写一个字的数据到指定的 I/O 端口
KP1812_CheckEvent	检查是否有设定的事件发生
KP1812_EnableEvent	启动或停止事件机制
KP1812_GetFIFOSize	获取设备 FIFO 大小
KP1812_FAIIntStart	开始单个通道的模拟量采集
KP1812_FAIIntScanStart	开始多个通道的模拟量采集
KP1812_FAITransfer	把采集到的数据传输到指定的缓存
KP1812_FAICheck	检查模拟量输入的状态
KP1812_ClearOverrun	清除 Overrun 标志
KP1812_FAITerminate	停止当前模拟量输入
KP1812_GetErrorMessage	根据错误代码返回错误信息

1、设备操作

a. KP1812_SelectDevice

Visual C++:

LRESULT KP1812_SelectDevice(ULONG ulDeviceNum);

Visual Basic:

Declare Function KP1812_SelectDevice Lib "KPCI1812.dll" (ulDeviceNum As Long) As Long

功能：该函数选择要操作的设备。
参数：ulDeviceNum：是 Device Number。
返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

b. KP1812_SelectDeviceFromList

Visual C++:

LRESULT KP1812_SelectDeviceFromList(HWND hCaller)

Visual Basic:

Declare Function KP1812_SelectDeviceFromList Lib "KPCI1812.dll"(ByVal hCaller As Long) As Long

功能：该函数从列表中选择要操作的设备。
参数：hCaller：调用该函数的窗口句柄。
返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

c. KP1812_DeviceGetFeatures

Visual C++:

LRESULT KP1812_DeviceGetFeatures(LPT_DeviceGetFeatures lpDevFeatures)

Visual Basic:

Declare Function KP1812_DeviceGetFeatures Lib "KPCI1812.dll" (lpDevFeatures As PT_DeviceGetFeatures) As Long

功能：该函数获取设备特征。
参数：lpDevFeatures：是 PT_DeviceGetFeatures 结构的指针，该结构包含了设备特征参数，结构定义请参照数据结构说明。
返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

d. KP1812_DeviceGetProperty

Visual C++:

LRESULT KP1812_DeviceGetProperty(USHORT nID, PVOID pData, ULONG* pDataLength);

Visual Basic:

Declare Function KP1812_DeviceGetProperty Lib "KPCI1812.dll" (ByVal nID As Integer, ByRef pData As Any, ByRef pDataLength As Long) As Long

功能：该函数获取设备指定的属性。
参数：nID：指属性类型；pData：指要获取属性的 buffer，lDataLength：指 buffer 大小。
返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

e. KP1812_DeviceGetProperty

Visual C++:

LRESULT KP1812_DeviceSetProperty(USHORT nID, PVOID pData, ULONG lDataLength);

Visual Basic:

Declare Function KP1812_DeviceSetProperty Lib "KPCI1812.dll" (ByVal nID As Integer, ByRef pData As Any, ByVal lDataLength As Long) As Long

功能：该函数设置设备指定的属性。
参数：nID：指属性类型；pData：指要设置属性的 buffer，lDataLength：指 buffer 大小。
返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

2、模拟量输入操作

a. KP1812_AIConfig

Visual C++:

LRESULT KP1812_AIConfig (LPT_AIConfig lpConfig);

Visual Basic:

Declare Function KP1812_AIConfig Lib "KPCI1812.dll" (ByRef lpAIConfig As PT_AIConfig) As Long

功能：该函数配置指定的单个模拟量输入通道。

参数：lpConfig：是 PT_AIConfig 结构的指针，该结构包含模拟量输入通道配置的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回回错误代码。

b. KP1812_AIGetConfig

Visual C++:

LRESULT KP1812_AIGetConfig (LPT_AIGetConfig lpAIGetConfig);

Visual Basic:

Declare Function KP1812_AIGetConfig Lib "KPCI1812.dll" (ByRef lpAIGetConfig As PT_AIGetConfig) As Long

功能：该函数获取指定的单个模拟量输入通道的配置

参数：lpAIGetConfig：是 PT_AIGetConfig 结构的指针，该结构包含要模拟量输入通道配置的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回回错误代码。

c. KP1812_AIBinaryIn

Visual C++:

LRESULT KP1812_AIBinaryIn (LPT_AIBinaryIn lpAIBinaryIn);

Visual Basic:

Declare Function KP1812_AIBinaryIn Lib "KPCI1812.dll" (ByRef lpAIBinaryIn As PT_AIBinaryIn) As Long

功能：该函数读取指定的单个通道模拟量输入的二进制值

参数：lpAIBinaryIn：是 PT_AIBinaryIn 结构的指针，该结构包含要获取输入的通道、二进制值等的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回回错误代码。

d. KP1812_AIScale

Visual C++:

LRESULT KP1812_AIScale (LPT_AIScale lpAIScale);

Visual Basic:

Declare Function KP1812_AIScale Lib "KPCI1812.dll" (lpAIScale As PT_AIScale) As Long

功能：该函数转换读取到的二进制值为电压值

参数：lpAIScale：是 PT_AIScale 结构的指针，该结构包含转换需要的参数和转换后的数值，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回回错误代码。

e. KP1812_AIVoltageIn

Visual C++:

LRESULT KP1812_AIVoltageIn (LPT_AIVoltageIn lpAIVoltageIn);

Visual Basic:

Declare Function KP1812_AIVoltageIn Lib "KPCI1812.dll" (lpAIVoltageIn As PT_AIVoltageIn) As Long

功能：该函数读取模拟量输入的电压值

参数：lpAIVoltageIn：是 PT_AIVoltageIn 结构的指针，该结构包含单个模拟量通道电压输入的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

f. KP1812_MAIConfig

Visual C++:

LRESULT KP1812_MAIConfig (LPT_MAIConfig lpMAIConfig);

Visual Basic:

Declare Function KP1812_MAIConfig Lib "KPCI1812.dll" (lpMAIConfig As PT_MAIConfig) As Long

功能：该函数配置指定的多个模拟量输入通道

参数：lpMAIConfig：是 PT_MAIConfig 结构的指针，该结构包含配置多个模拟量通道的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

g. KP1812_MAIBinaryIn

Visual C++:

LRESULT KP1812_MAIBinaryIn (LPT_MAIBinaryIn lpMAIBinaryIn);

Visual Basic:

Declare Function KP1812_MAIBinaryIn Lib "KPCI1812.dll" (lpMAIBinaryIn As PT_MAIBinaryIn) As Long

功能：该函数读取指定的多个通道模拟量输入的二进制值

参数：lpMAIBinaryIn：是 PT_MAIBinaryIn 结构的指针，该结构包含要获取输入的起始通道、通道数以及二进制值数组等的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

h. KP1812_MAIVoltageIn

Visual C++:

LRESULT KP1812_MAIVoltageIn (LPT_MAIVoltageIn lpMAIVoltageIn);

Visual Basic:

Declare Function KP1812_MAIVoltageIn Lib "KPCI1812.dll" (lpMAIVoltageIn As PT_MAIVoltageIn) As Long

功能：该函数读取多个模拟量通道输入的电压

参数：lpMAIVoltageIn：是 PT_MAIVoltageIn 结构的指针，该结构包含多个模拟量通道同时电压输入的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

3、I/O 端口操作

a. KP1812_ReadPortByte

Visual C++:

LRESULT KP1812_ReadPortByte(LPT_ReadPortByte lpReadPortByte);

Visual Basic:

Declare Function KP1812_ReadPortByte Lib "KPCI1812.dll" (lpReadPortByte As PT_ReadPortByte) As Long

功能：该函数读一个字节的 I/O 端口数据

参数：lpReadPortByte：是 PT_ReadPortByte 结构的指针，该结构包含读取一字节端口数据的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

b. KP1812_WritePortByte

Visual C++:

LRESULT KP1812_WritePortByte(LPT_WritePortByte lpWritePortByte);

Visual Basic:
Declare Function KP1812_WritePortByte Lib "KPCI1812.dll" (lpWritePortByte As PT_WritePortByte) As Long

功能：该函数写一个字节数据到 I/O 端口

参数：lpWritePortByte：是 PT_WritePortByte 结构的指针，该结构包含写一个字节数据到端口的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

c. KP1812_ReadPortWord

Visual C++:
LRESULT KP1812_ReadPortWord(LPT_ReadPortWord lpReadPortWord);

Visual Basic:
Declare Function KP1812_ReadPortWord Lib "KPCI1812.dll" (lpReadPortWord As PT_ReadPortWord) As Long

功能：该函数从 I/O 端口读一个字的数据

参数：lpReadPortWord：是 PT_ReadPortWord 结构的指针，该结构包含从指定的 I/O 端口读取一个字数据的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

d. KP1812_WritePortWord

Visual C++:
LRESULT KP1812_WritePortWord(LPT_WritePortWord lpWritePortWord);

Visual Basic:
Declare Function KP1812_WritePortWord Lib "KPCI1812.dll" (lpWritePortWord As PT_WritePortWord) As Long

功能：该函数写一个字节数据到 I/O 端口

参数：lpWritePortWord：是 PT_WritePortWord 结构的指针，该结构包含写一个字的数据到指定 I/O 端口的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

4、高速数据采集操作

a. KP1812_CheckEvent

Visual C++:
LRESULT KP1812_CheckEvent (LPT_CheckEvent lpCheckEvent);

Visual Basic:
Declare Function KP1812_CheckEvent Lib "KPCI1812.dll" (lpCheckEvent As PT_CheckEvent) As Long

功能：该函数检查是否有设定的事件发生

参数：lpCheckEvent：是 PT_CheckEvent 结构的指针，该结构包含检测事件状态的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

b. KP1812_EnableEvent

Visual C++:
LRESULT KP1812_EnableEvent (LPT_EnableEvent lpEnableEvent);

Visual Basic:
Declare Function KP1812_EnableEvent Lib "KPCI1812.dll" (lpEnableEvent As PT_EnableEvent) As Long

功能：该函数启动或停止事件机制

参数：lpEnableEvent：是 PT_EnableEvent 结构的指针，该结构包含使能事件状态的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；

如果未成功，则返回错误代码。

c. KP1812_GetFIFOSize

Visual C++:

LRESULT KP1812_GetFifoSize(PULONG lSize);

Visual Basic:

Declare Function KP1812_GetFifoSize Lib "KPCI1812.dll" (lSize As Long) As Long

功能：该函数获取设备 FIFO 大小

参数：lSize：是 long 的指针，该参数返回 FIFO 的大小。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

d. KP1812_FAIIntStart

Visual C++:

LRESULT KP1812_FAIIntStart (LPT_FAIIntStart lpFAIIntStart);

Visual Basic:

Declare Function KP1812_FAIIntStart Lib "KPCI1812.dll" (lpFAIIntStart As PT_FAIIntStart) As Long

功能：该函数开始单个通道的模拟量采集

参数：lpFAIIntStart：是 PT_FAIIntStart 结构的指针，该结构包含启动单通道模拟量数据采集的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

e. KP1812_FAIIntScanStart

Visual C++:

LRESULT KP1812_FAIIntScanStart (LPT_FAIIntScanStart lpFAIIntScanStart);

Visual Basic:

Declare Function KP1812_FAIIntScanStart Lib "KPCI1812.dll" (lpFAIIntScanStart As PT_FAIIntScanStart) As Long

功能：该函数开始多个通道的模拟量采集

参数：lpFAIIntScanStart：是 PT_FAIIntScanStart 结构的指针，该结构包含启动多个通道模拟量数据采集的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

f. KP1812_FAITransfer

Visual C++:

LRESULT KP1812_FAITransfer (LPT_FAITransfer lpFAITransfer);

Visual Basic:

Declare Function KP1812_FAITransfer Lib "KPCI1812.dll" (lpFAITransfer As PT_FAITransfer) As Long

功能：该函数把采集到的数据传输到指定的缓存

参数：lpFAITransfer：是 PT_FAITransfer 结构的指针，该结构包含传输数据到缓存的参数，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

g. KP1812_FAICheck

Visual C++:

LRESULT KP1812_FAICheck (LPT_FAICheck lpFAICheck);

Visual Basic:

Declare Function KP1812_FAICheck Lib "KPCI1812.dll" (lpFAICheck As PT_FAICheck) As Long

功能：该函数检查模拟量输入的状态

参数：lpFAICheck：是 PT_FAICheck 结构的指针，该结构包含一些状态的指针，结构定义请参照数据结构说明。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

h. KP1812_ClearOverrun

Visual C++:

LRESULT KP1812_ClearOverrun(VOID);

Visual Basic:

Declare Function KP1812_ClearOverrun Lib "KPCI1812.dll" () As Long

功能：该函数清除 Overrun 标志

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

i. KP1812_FAITerminate

Visual C++:

LRESULT KP1812_FAITerminate (VOID);

Visual Basic:

Declare Function KP1812_FAITerminate Lib "KPCI1812.dll" () As Long

功能：该函数停止当前模拟量输入

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

5、其他操作

a. KP1812_GetErrorMessage

Visual C++:

LRESULT KP1812_GetErrorMessage(LRESULT lError, TCHAR* lpMsg);

Visual Basic:

Declare Function KP1812_GetAddress Lib "KPCI1812.dll" (lpVoid As Any) As Long

功能：该函数根据错误代码返回错误信息

参数：lError：错误代码，lpMsg：指向错误信息字符串的指针。

返回值：如果执行成功，则返回 0；
如果未成功，则返回错误代码。

四、数据结构说明

1、设备操作

a) PT_DeviceGetFeatures

```
typedef struct tagPT_DeviceGetFeatures
{
    LPDEVFEATURES buffer;
    USHORT size;
}PT_DeviceGetFeatures, FAR * LPT_DeviceGetFeatures;
```

Member Description:

名称	方向	类型	描述
buffer	Output	**DEVFEATURES** 的指针	指向设备特征结构的指针
size	Input	USHORT	特征结构的数据长度

b）DEVFEATURES

```
typedef struct tagDEVFEATURES
{
    USHORT    usMaxAIDiffChl;
    USHORT    usMaxAISiglChl;
```

```
        USHORT     usMaxTimerChl;
        USHORT     usMaxAlarmChl;
        USHORT     usNumADBit;
        USHORT     usNumADByte;
        USHORT     usNumGain ;
        GAINLIST glGainList[32];
        DWORD       dwPermutation[4];
 } DEVFEATURES, FAR * LPDEVFEATURES;
```

Member Description:

名称	方向	类型	描述
usMaxAIDiffChl	Output	USHORT	差分模拟量输入通道的最大个数
usMaxAISiglChl	Output	USHORT	单端模拟量输入通道的最大个数
usMaxTimerChl	Output	USHORT	计数器/时钟的最大个数
usMaxAlarmChl	Output	USHORT	报警通道的最大个数
usNumADBit	Output	USHORT	A/D 转换的位数
usNumADByte	Output	USHORT	A/D 转换所需字节个数
usNumGain	Output	USHORT	输入范围的最大个数
glGainList	Output	**GAINLIST** 数组	所有支持输入范围
dwPermutation	Output	DWORD	设备所支持功能

Note：

dwPermutation 的定义：

Bit 0: Software AI
Bit 1: DMA AI
Bit 2: Interrupt AI
Bit 3: Condition AI
Bit 4: Software AO
Bit 5: DMA AO
Bit 6: Interrupt AO
Bit 7: Condition AO
Bit 8: Software DI
Bit 9: DMA DI
Bit 10: Interrupt DI
Bit 11: Condition DI
Bit 12: Software DO
Bit 13: DMA DO
Bit 14: Interrupt DO
Bit 15: Condition DO
Bit 16: High Gain
Bit 17: Auto Channel Scan
Bit 18: Pacer Trigger
Bit 19: External Trigger
Bit 20: Down Counter
Bit 21: Dual DMA
Bit 22: Monitoring
Bit 23: Qcounter

c) GAINLIST

```
typedef struct tagGAINLIST
{
    USHORT usGainCde;
    FLOAT fMaxGainVal;
    FLOAT fMinGainVal;
    CHAR szGainStr[16];
} GAINLIST;
```

Member Description:

名称	方向	类型	描述

usGainCde	Output	USHORT	该模拟量输入范围的代码
fMaxGainVal	Output	float	该模拟量输入范围的最大值
fMinGainVal	Output	float	该模拟量输入范围的最小值
szGainStr	Output	Char 数组	该模拟量输入范围的字符描述

2、模拟量输入操作

a) PT_AIConfig

```
typedef struct tagPT_AIConfig
{
USHORT DasChan;
USHORT DasGain;
} PT_AIConfig, * LPT_AIConfig;
```

Member Description:

名称	方向	类型	描述
DasChan	Input	USHORT	采样通道
DasGain	Input	USHORT	采样值范围的代码

b) PT_AIGetConfig

```
typedef struct tagPT_AIGetConfig
{
LPDEVCONFIG_AI buffer;
USHORT size;
} PT_AIGetConfig, * LPT_AIGetConfig;
```

Member Description:

名称	方向	类型	描述
buffer	Output	**DEVONFIG_AI** 指针	指向设备配置结构的指针
size	Input	USHORT	该配置结构的大小

c) PT_AIBinaryIn

```
typedef struct tagPT_AIBinaryIn
{
USHORT chan;
USHORT TrigMode;
USHORT *reading;
} PT_AIBinaryIn, * LPT_AIBinaryIn;
```

Member Description:

名称	方向	类型	描述
chan	Input	USHORT	采样通道
TrigMode	Input	USHORT	触发模式。0 为内部触发；1 为外部触发
reading	Output	USHORT 指针	从采样通道读取的原始数据

d) PT_AIScale

```
typedef struct tagPT_AIScale
{
USHORT reading;
FLOAT MaxVolt;
USHORT MaxCount;
USHORT offset;
FLOAT   *voltage;
} PT_AIScale, * LPT_AIScale;
```

Member Description:

名称	方向	类型	描述
reading	Input	USHORT	原始数据
MaxVolt	Input	float	该输入范围的最大电压值
MaxCount	Input	USHORT	最大范围值（4095）
offset	Input	USHORT	0 伏特电压的偏移
voltage	Output	（float）浮点数指针	转换后的电压值（伏特）

e) PT_AIVoltageIn

```
typedef struct tagPT_AIVoltageIn
{
USHORT chan;
USHORT gain;
USHORT TrigMode;
FLOAT    *voltage;
} PT_AIVoltageIn, * LPT_AIVoltageIn;
```

Member Description:

名称	方向	类型	描述
chan	Input	USHORT	采样通道
gain	Input	USHORT	该模拟量输入范围代码
TrigMode	Input	USHORT	触发模式
voltage	Output	（float）浮点数指针	输入的电压,已转换成伏特值的形式

f) PT_MAIConfig

```
typedef struct tagPT_MAIConfig
{
USHORT NumChan;
USHORT StartChan;
USHORT *GainArray;
} PT_MAIConfig, * LPT_MAIConfig;
```

Member Description:

名称	方向	类型	描述
NumChan	Input	USHORT	要配置的通道数
StartChan	Input	USHORT	起始通道
GainArray	Input	USHORT 指针	模拟量输入范围代码的数组

g) PT_MAIBinaryIn

```
typedef struct tagPT_MAIBinaryIn
{
USHORT NumChan;
USHORT StartChan;
USHORT TrigMode;
USHORT *ReadingArray;
} PT_MAIBinaryIn, * LPT_MAIBinaryIn;
```

Member Description:

名称	方向	类型	描述
NumChan	Input	USHORT	采样的通道数
StartChan	Input	USHORT	采样的起始通道
TrigMode	Input	USHORT	触发模式
ReadingArray	Output	USHORT 指针	模拟量输入的原始数据数组

h) PT_MAIVoltageIn

```
typedef struct tagPT_MAIVoltageIn
{
USHORT NumChan;
USHORT StartChan;
USHORT *GainArray;
USHORT TrigMode;
FLOAT    *VoltageArray;
} PT_MAIVoltageIn, * LPT_MAIVoltageIn;
```

Member Description:

名称	方向	类型	描述
NumChan	Input	Unsigned short	采样的通道数
StartChan	Input	USHORT	采样的起始通道
GainArray	Input	USHORT 指针	各个通道的模拟量输入范围代码
TrigMode	Input	USHORT	触发模式
VoltageArray	Output	（float）浮点数指针	模拟量输入的电压数据数组

3、I/O 端口操作

a) PT_ReadPortByte

```
typedef struct tagPT_ReadPortByte
{
USHORT port;
USHORT *ByteData;
} PT_ReadPortByte, * LPT_ReadPortByte;
```

Member Description:

名称	方向	类型	描述
port	Input	USHORT	端口地址
ByteData	Output	USHORT 指针	从端口读取的字节数据

b) PT_WritePortByte

```
typedef struct tagPT_WritePortByte
{
USHORT port;
USHORT ByteData;
} PT_WritePortByte, * LPT_WritePortByte;
```

Member Description:

名称	方向	类型	描述
port	Input	USHORT	端口地址
ByteData	Input	USHORT 指针	写入端口的字节数据

c) PT_ReadPortWord

```
typedef struct tagPT_ReadPortWord
{
USHORT port;
USHORT *WordData;
} PT_ReadPortWord, * LPT_ReadPortWord;
```

Member Description:

名称	方向	类型	描述
port	Input	USHORT	端口地址
WordData	Output	USHORT 指针	从端口读取的一个字的数据

d) PT_WritePortWord

```
typedef struct tagPT_WritePortWord
{
USHORT port;
USHORT WordData;
} PT_WritePortWord, * LPT_WritePortWord;
```

Member Description:

名称	方向	类型	描述
port	Input	USHORT	端口地址
WordData	Input	USHORT 指针	写入端口的一个字的数据

4、高速数据采集操作

a) PT_CheckEvent

```
typedef struct tagPT_CheckEvent
{
USHORT *EventType;
DWORD Milliseconds;
} PT_CheckEvent, * LPT_CheckEvent;
```

Member Description:

名称	方向	类型	描述
EventType	Output	DWORD	驱动支持的事件类型，请参考 **PT_EnableEvent**
Milliseconds	Input	USHORT 指针	事件超时的毫秒数

b) PT_EnableEvent

```
typedef struct tagPT_EnableEvent
{
USHORT EventType;
USHORT Enabled;
USHORT Count;
} PT_EnableEvent, * LPT_EnableEvent;
```

Member Description:

名称	方向	类型	描述
EventType	Input	USHORT	希望触发的事件类型
Enabled	Input	USHORT	使能或取消触发的事件类型。1 为使能；0 为取消
Count	Input	USHORT	希望事件触发的次数

注意：

EventType 的定义：

0 位：中断事件

1 位：buffer change 事件

2 位：termination 事件

3 位：overrun 事件

c) PT_FAIIntStart

```
typedef struct tagPT_FAIIntStart
{
USHORT TrigSrc;
DWORD SampleRate;
USHORT chan;
USHORT gain;
```

```
USHORT *buffer;
ULONG count;
USHORT cyclic;
USHORT IntrCount;
} PT_FAIIntStart, * LPT_FAIIntStart;
```

Member Description:

名称	方向	类型	描述
TrigSrc	Input	USHORT	触发源: 1 位外部触发源, 0 位内部触发源
SampleRate	Input	DWORD	采样频率
chan	Input	USHORT	采样通道
gain	Input	USHORT	该通道输入范围代码
buffer	Output	USHORT 指针	用户分配的缓存
count	Input	ULONG	采样的次数
cyclic	Input	USHORT	是否循环模式: 1 位循环模式, 0 位非循环模式
IntrCount	Input	USHORT	采样多少次一中断,（只有 FIFO 和中断两种采样模式，中断为 1，FIFO 则为半满 FIFO 大小）

d) PT_FAIIntScanStart

```
typedef struct tagPT_FAIIntScanStart
{
USHORT TrigSrc;
DWORD SampleRate;
USHORT NumChans;
USHORT StartChan;
USHORT *GainList;
USHORT *buffer;
ULONG count;
USHORT cyclic;
USHORT IntrCount;
} PT_FAIIntScanStart, * LPT_FAIIntScanStart;
```

Member Description:

名称	方向	类型	描述
TrigSrc	Input	USHORT	触发源: 1 位外部触发源, 0 位内部触发源
SampleRate	Input	DWORD	采样频率
NumChans	Input	USHORT	采样通道数
StartChan	Input	USHORT	采样起始通道
GainList	Input	USHORT 指针	各个通道输入范围代码
buffer	Output	USHORT 指针	用户分配的缓存
count	Input	ULONG	采样的次数
cyclic	Input	USHORT	是否循环模式: 1 位循环模式, 0 位非循环模式

IntrCount	Input	USHORT	采样多少次一中断，（只有 FIFO 和中断两种采样模式，中断为 1，FIFO 则为半满 FIFO 大小）

e) PT_FAITransfer

```
typedef struct tagPT_FAITransfer
{
USHORT ActiveBuf;
PVOID DataBuffer;
USHORT DataType;
ULONG start;
ULONG count;
USHORT *overrun;
} PT_FAITransfer, * LPT_FAITransfer;
```

Member Description:

名称	方向	类型	描述
ActiveBuf	Input	USHORT	Buffer 的类型，本设备总是为 0
DataBuffer	Output	指向浮点数或 USHORT 的指针	如果 buffer 指针不为空，当数据类型为 USHORT，则该 buffer 的长度应该不小于 **2*count**。当数据类型为浮点数时，则该 buffer 的长度应该不小于 **4*count**。
DataType	Input	USHORT	数据类型，0 为原始数据，1 为电压值（浮点数）
start	Input	ULONG	从源 buffer 复制到用户 buffer 的起始点
count	Input	ULONG	从源 buffer 复制到用户 buffer 的采样个数
overrun	Output	ULONG 指针	overrun 状态 0- 无 overrun 发生 1- Overrun 发生了

f) PT_FAICheck

```
typedef struct tagPT_FAICheck
{
USHORT far  *ActiveBuf;
USHORT far  *stopped;
ULONG  far  *retrieved;
USHORT far  *overrun;
USHORT far  *HalfReady;
} PT_FAICheck, * LPT_FAICheck;
```

Member Description:

名称	方向	类型	描述
ActiveBuf	Output	USHORT 指针	本设备总是为 0
stopped	Output	USHORT 指针	说明操作是否完成，0 代表未完成，1 代表完成 。
retrieved	Output	ULONG 指针	A/D 转换的次数
overrun	Output	USHORT 指针	是否有 overrun 发生， 0 无 overrun， 1 有 overrun。

HalfReady	Output	USHORT 指针	说明 buffer 状态， 0 为数据没准备好； 1 为第一次半满 buffer 准备. 2 为第二次半满 buffer 准备。

第五章　KPCI-1812 A/D卡的校准、保修和注意事项

一　应用注意事项 ：

在公司售出的产品包装中，用户将会在资料光盘中找到本卡的说明书，同时还有产品质保卡。产品质保卡请用户务必妥善保存，当该产品出现问题需要维修时，请与本公司联系；并将产品质保卡同产品一起，寄回本公司，以便我们能尽快的帮用户解决问题。

在使用KPCI-1812板时，应注意以下问题：

① KPCI-1812卡正面的IC芯片不要用手去摸，防止芯片受到静电的损害。

② 在使用KPCI-1812卡时，可通过K-801E等信号调理端子板与现场信号连接。

③ 用户务必注意电源的开关顺序，使用时要求先开主机电源，后开信号源的电源；先关信号源的电源，后关主机电源。

二、校准：

KPCI-1812板出厂时已经校准，只有当用户使用一段时间后，或者用户认为需要时才做校准。下面以0～10V量程为例，说明校准过程：

准备一块4位半精度以上数字电压表，安装好KPCI-1812，打开主机电源，预热10分种。

A/D的校准

（一） 单极性输出的校准

1） 将卡上的信号增益短路套设定为1倍。

2） ① 零点调整

选模拟输入的任意两个通道，比如Ain1、Ain2通道，将Ain1通道输入接0伏电压(AGND),在WINDOWS下运行KPCI-1812 测试程序,选择1通道，选择单通道数据采集方式，采集开始后，调整电位器VR1，使显示值为 0。

② 满度调整

将 Ain2接正满度电压+9998 mV，在选择2通道，采集程序开始采集后，调整电位器VR2，使显示值为 10.000V。

重复以上步骤，直到满足要求为止。

3） 如果放大器增益选择其他倍数，在相应的通道上，也要接入相应的满度电压。（5V或其他值）

三、保修：

KPCI-1812自出厂之日起，两年内凡用户遵守贮存，运输和使用要求，而产品质量低于技术指标的，凭保修卡免费维修。因违反操作规定和要求而造成损坏的，需交纳器件维修费。

四、 产品配套清单：

4.1 KPCI-1812 32通道模拟量数据采集卡壹块。
4.2 北京科瑞兴业公司产品光盘壹张。
4.3 37芯D型插头壹套。